mlBibTeX Architecture

Jean-Michel Hufflen

MlBibTeX's Architecture

Jean-Michel HUFFLEN

Abstract

This paper shows the architecture of MlBibTeX, our reimplementation of BibTeX focusing on multilingual features. Making precise the organisation and modules of this architecture allows us to show how MlBibTeX works and focus on the differences between MlBibTeX and BibTeX from a conceptual point of view. We also explain why this implementation using the Scheme programming language allows users to scrutinise the result of intermediate steps.

Keywords LaTeX, BibTeX, MlBibTeX, bibliography files, bibliography styles, bst, nbst, XML, XSLT.

Sommario

Questo paper illustra l'architettuta di MlBibTeX, una reimplementazione di BibTeX più adatta per lavorare in un contesto multilingua. Rendendo più precisa l'organizzazione ed i moduli è possibile mostrare come MlBibTeX lavora evidenziando le principali differenze esistenti tra MlBibTeX e BibTeX da un punto di vista concettuale. Verrà inoltre spiegato perché quest'implementazione fatta usando il linguaggio di programmazione Scheme permette di verificare i risultati degli step intermedi.

Parole chiave LaTeX, BibTeX, MlBibTeX, bibliografia, stile bibliografico, bst, nbst, XML, XSLT.

0 Introduction

Often the LaTeX [15] word processor and BibTeX [17] bibliography processor are used in conjunction. Such an approach allows the 'References' section of a printed document to be given nice layout. Homogeneous typesetting for items being same type is provided by means of bibliography styles ruling the layout of articles, books, etc. As mentioned in [16, § 13.1], BibTeX has been stable for a long time. However, modern features, such as multilingual capabilities, have not been incorporated and are possible only by means of workarounds, most often by inserting LaTeX commands inside field values. As a consequence, using BibTeX to generate bibliographies for another output format than LaTeX may be difficult.

There are several projects aiming to develop BibTeX's successors [16, § 13.1.2]. Among them, MlBibTeX—for '**M**ulti**L**ingual BibTeX—focuses on multilingual features. It uses XML[1] as a central formalism. Even if there is a compatibility mode for bibliography styles originating from BibTeX [10], style designers are encouraged to develop new styles using a new language [4, App. A & B], nbst[2], close to XSLT[3] [21], the language of transformations used for XML texts.

The broad outlines of MlBibTeX have already been shown at the 2004 GUIT[4] conference and given in [9]. This present paper focuses on MlBibTeX's architecture, in the sense that we explain how MlBibTeX's modules are organised and how they interact. First, we recall how BibTeX proceeds. So we can point out the differences between BibTeX and MlBibTeX from a conceptual point of view. Reading this article only requires basic knowledge about LaTeX, BibTeX and XML. About its implementation, MlBibTeX is written using the Scheme programming language [8]. We will not go thoroughly into technical points in Scheme[5], but mention how intermediate results are built.

1 Situation

1.1 BibTeX's behaviour

As a recent reference, [16, § 12.1.3] explains how LaTeX and BibTeX can cooperate. *Citation keys*—e.g., '`eco1980`' in '`\cite{eco1980}`'—are stored by LaTeX into an auxiliary (.aux file) also used by BibTeX. Now let us describe BibTeX's behaviour roughly. First, it takes an .aux file[6] and searches it for citations. This .aux file also contains the information about a bibliography style. Then this bibliography style file is run. If we look at some standard bibliography files, some variables, functions and macros are defined, bibliographic databases are searched for citation keys, and the accurate functions of the style are applied. Figure 1 describes the organisation of a bibliography style file as it is sketched in [16, § 13.6.2]. To sum up: once the citation keys and the bibliography style are known, the latter is run. The result is a .bbl file containing a `thebibliography` environment [16,

1. e**X**tensible **M**arkup **L**anguage. Readers interested in an introductory book to this formalism can refer to [18].
2. **N**ew **B**ibliography **ST**yles.
3. e**X**tensible **S**tylesheet **L**anguage **T**ransformations.
4. ***G**ruppo **U**tilizzatori **I**taliani di TEX* (Italian TEX user group).
5. The current reference manual of this language is [13].
6. Let us recall that when BibTeX is invoked by means of a command line such as '`bibtex` ⟨*job-name*⟩', ⟨*job-name*⟩ refers to an .aux file.

```
ENTRY {...}{...}{...}      % Declaration of variables and functions.
FUNCTION {...}{...}
...
FUNCTION {article}{...}
...
MACRO {oct}{"October"}
...
READ                       % Search bibliography data base .bib files for entries.
...                        % Some preprocessing, including the sort of selected entries.
SORT
...
EXECUTE {begin.bib}        % Preamble and \begin{thebibliography}.
EXECUTE {...}              % Some initialisations.
ITERATE {call.type$}       % Loop over entries producing outputs.
EXECUTE {end.bib}          % Write \end{thebibliography} command.
```

FIGURE 1: Bibliography style file used by BibTEX: framework and basic flow.

§ 12.1.2], each reference being prefixed by means of the `\bibitem` command.

1.2 Going further

At a first glance, a successor of BibTEX might conserve the same *modus operandi*. That is not always possible. BibTEX never reads the .tex file because the information it needs can be found in .aux files only. The same behaviour is unsuitable for MlBibTEX—as a program managing multilingual bibliographies—when it generates a 'References' section for a LaTeX source text. Let us consider the following title, which uses a syntactical extension provided by MlBibTEX:

```
TITLE = {[Danse macabre] : french}
```

This book of Stephen King is written in English—of course—but its title uses French words, what is specified by the '`[...] : french`' notation, where '`french`' is a *language identifier* [4]. Let us assume that we are building a 'References' section for a document in English. In order for LaTeX to use the right hyphenation patterns if need be, the .bbl file generated should include such a title as:

```
\foreignlanguage{french}{Danse macabre}
```

provided that the babel package is loaded with the french option when LaTeX processes this document [16, Ch. 9]. First, there are several ways to write in French with LaTeX with the babel package or with other *ad hoc* packages. Second, let us recall that all the languages used throughout a document must be specified as options when the babel package is loaded. This package is static and the languages it uses cannot be extended dynamically. As we explain in [7], there are several solutions to this problem, but we think that the best consists of analysing the preamble of the .tex file, in order to get information about the multilingual capabiblity chosen by a user. Let us go back to our title: if the french option has not been selected, MlBibTEX will just produce '`Danse macabre`' and warn the user that some words may be incorrectly hyphenated.

Another drawback of bibliography styles of BibTEX is that they are monolithic: one file for one style. There is a tool, `custom-bib` [16, § 13.5.2], that allows a bibliography style to be build automatically from users' wishes. But there is no modular approach in the sense that a style designer could develop a new style from existing style fragments. In MlBibTEX, a style is assembled dynamically, according to what is needed. For example, let us consider that a plain style[7] is to be used for a document written in English and including some fragments in Italian. The files that can be used are:

plain.nbst general definitions for this style.

plain-{english,italian}.nbst definitions suitable for the English and Italian languages and belonging to this style.

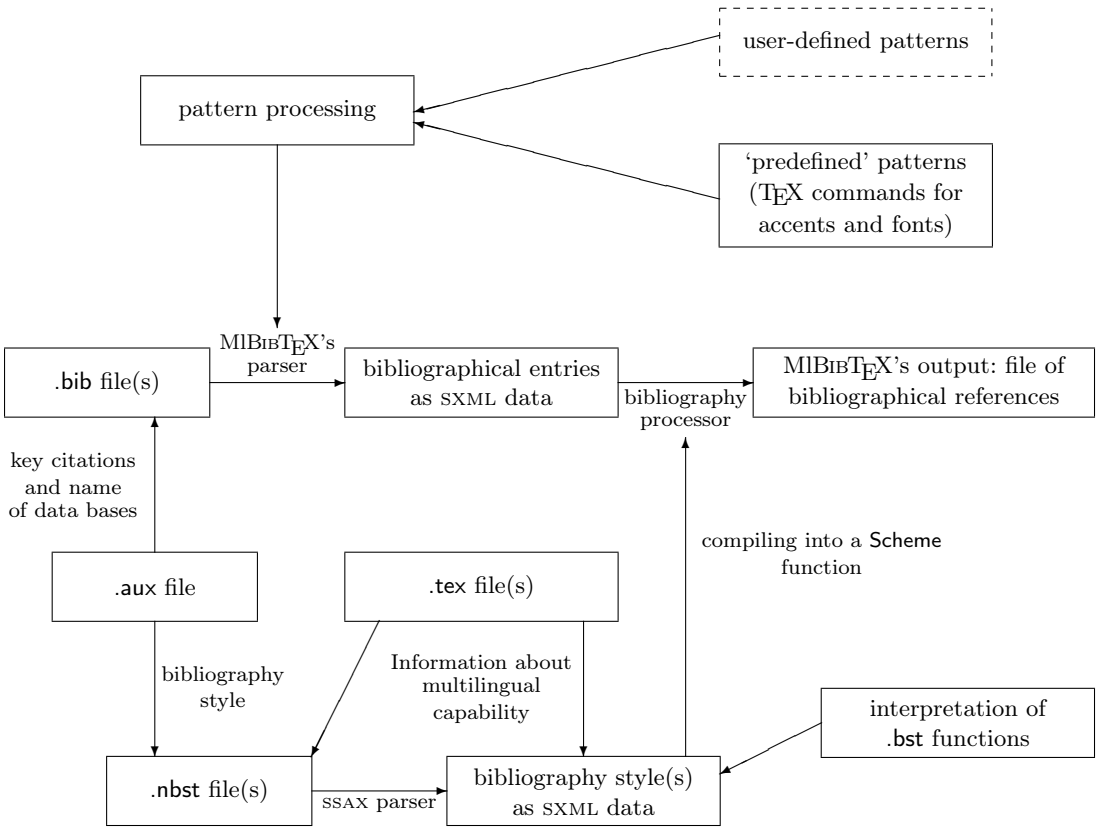
{-english,-italian}.nbst general definitions for the English and Italian languages.

So we need this information about available languages before applying a style.

There is another difference, which can be noticed by an end-user, about abbreviations usable in a .bib file. Some may be defined using the `@STRING{...}` command of BibTEX [16, § 13.2.3], some may be defined as macros within the bibliography style [16, Table 13.7], for example, month names:

```
MONTH = oct
```

7. That is, items are labelled by numbers.



'$A \leftarrow B$' means that A uses B. More precisely, functions or data put in A use functions of B or data from B. A dashed box means that the corresponding module is planned, but not fully implemented yet.

FIGURE 2: Data flow in MlBibTeX.

(cf. Figure 1). When BibTeX reads a .bib file, all the abbreviations must be expanded into strings. That is, all must be known at this time. On the contrary, in the styles of MlBibTeX, you can perform different operations whether or not a symbol is defined:

```
<nbst:choose>
  <nbst:when test="is-defined('roma')">
    ...
    <!--  The is-defined function does not exist
          in XSLT, but exists in nbst.
      -->
  </nbst:when>
  ...
</nbst:choose>
```

We think that all these differences between MlBibTeX and BibTeX are improvements but they lead to a redesign of the data flow in MlBibTeX.

2 Organisation of MlBibTeX

The organisation of MlBibTeX is pictured at Figure 2. Like BibTeX, MlBibTeX reads an .aux file and gets a list of citation keys. It also gets information about some bibliography data base files and a bibliography style.

Then it parses .bib files, searching them for the citation keys. This step results in an SXML[8] tree, that is, a representation of an XML tree using Scheme [14]. Let us consider that the entry given in Figure 3 is to be put within a bibliography. Conceptually, it can be viewed as the XML tree given in Figure 4, represented in SXML by the Scheme lists of Figure 5. Bibliography styles use XPath[9] expressions related to XML trees resulting from parsing a .bib file, that is, XML trees comparable with the example given in Figure 4.

Some symbols are not expanded because they have not been defined by a `@STRING` command in a .bib file—e.g., the `roma` symbol in Figure 3, see also Appendix A—and are retained for future processing. In other words, that is a kind of 'benefit of the doubt'. The commands generating accented letters and other signs belonging to the Latin 1 encoding are expanded: for exam-

8. Scheme implementation of XML.
9. XPath is the language used to address parts of an XML document [20].

```
@BOOK{eco1980,
        AUTHOR = {Umberto Eco},
        TITLE = {Il nome della rosa},
        PUBLISHER = {Bompiani},
        ADDRESS = roma,
        YEAR = 1980,
        LANGUAGE = italian}
```

Notice the `LANGUAGE` field and language identifier used to specify this entry for a book written in Italian.

FIGURE 3: MlBibTEX's bibliographical entry.

```
<book id="eco1980" language="italian">
  <author>
    <name>
      <personname>
        <first>Umberto</first>
        <last>Eco</last>
      </personname>
    </name>
  </author>
  <title>Il nome della rosa</title>
  <publisher>Bompiani</publisher>
  <year>1980</year>
  <address><symbol name="roma"/></address>
</book>
```

FIGURE 4: Entry as an XML tree (Fig. 3 cont'd).

ple, '`\'{e}`' is expanded to the accented letter '`é`' of this encoding[10]. As a consequence, the accented and special letters included in the values of MlBibTEX's fields are viewed as single letters when entries are sorted. Besides, this sort operation can be performed w.r.t. a language's lexicographical order. The nbst language provides a `nbst:sort` element [4, App. A] that is analogous to the `xsl:sort` element of XSLT [21, § 10]: if the data to be sorted are strings, a `language` attribute allows sorting w.r.t. a particular language (English, Italian, Spanish, ...)

Likewise, some predefined commands of LaTeX for fonts (`\emph`, `\textbf`, ...) are recognised and expanded to XML elements. This is done by means of patterns, expressed in Scheme. Now, only some predefined patterns are processed, e.g.:

```
(define-pattern "\'e" "é")
(define-pattern "\'{e}" "é")
```

we plan to extend this capability to user-defined patterns, so users will be able to mark up their .bib files, and tell MlBibTEX how it can understand this markup [5].

During the next step, MlBibTEX deduces the source .tex file's name from the .aux file's[11]. Then it parses the preamble of this file and try to know:

- which languages are used throughout this document and how;
- which encoding may be used: pure ASCII[12] or Latin 1. So, users can type accented letters directly inside field values of .bib files processed by MlBibTEX, even if they do not use the inputenc package of LaTeX, for dealing with encodings other than ASCII [16, § 7.5.2]. In this last case, the commands for generating accents will be used, for example, '`é`' will be transformed into '`\'{e}`'.

This step results in:

- a list of possible file names[13] whose union is the multilingual bibliography style;
- updating a set of Scheme functions controlling MlBibTEX's output.

Then the files of the bibliography style are parsed by means of SSAX[14] [14]. This parser returns the SXML representation of an XML tree. So, the result is comparable to the expression given in Figure 5, but concerns the elements of a bibliography style. Finally, the fragments of a bibliography style are assembled and compiled into a Scheme function[15] [12]. Then this function is applied to the list of bibliographical entries selected.

Last, bibliography styles may use functions written using the bst language of BibTEX. That allows 'old' bibliography styles to be replaced progressively and explains the arrow, in Figure 2, from the box concerning bst functions to the box concerning bibliography styles expressed using the new language nbst.

3 Conclusion

When we started to develop MlBibTEX's first version [2], we only aimed to provide a 'better BibTEX'. The only changes from end-users' point of view were additional syntactic features. This approach was not satisfactory enough and we decided to design a new language for bibliography styles [3], taking advantage of XML features as far as possible. This design choice and the implementation of multilingual features led to an

10. Now, most Scheme interpreters can only deal with the Latin 1 encoding. Future versions should be Unicode-compliant [19] and able to deal with more encodings, such as Latin 2, UTF-8, ...

11. You can specify this .tex file's name by means of an option of MlBibTEX, if need be.

12. **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange.

13. 'Possible files' because some files specific to a particular language may be inexistent, in which case they are ignored.

14. Scheme implementation of SAX (Simple API for XML).

15. Displaying a function's body is difficult and often not provided by most Scheme interpreters, that is why we do not give more details about this step. But we plan to develop tools for browsing bibliography styles.

```
(*top* ...
 (book (@ (id "eco1980") (language "italian"))
       (author
        (name (personname (first "Umberto")
                          (last "Eco"))))
       (title "Il nome della rosa")
       (publisher "Bompiani") (year "1980")
       (address (symbol (@ (name "roma"))))))
```

FIGURE 5: Entry in SXML (Fig. 4 cont'd).

architecture quite far from BibTeX's. It may appear as more complicated, but we think that the role of each module is precise and can be understood by end-users, provided that they have got first experience with the Scheme programming language. But this architecture seems to us to be stable and robust. In particular, we succeeded in using MlBibTeX for deriving 'References' sections for ConTeXt [11], another format built out of TeX[16] and very different from LaTeX. So, extending MlBibTeX to other output formats is possible.

A Abbreviations in MlBibTeX

This appendix aims to make precise how abbreviations—possibly multilingual—can be specified in MlBibTeX. As an example, we consider the `roma` symbol, that appears in the `eco1980` entry of Figure 3. Another example is given in [6].

You can use a `@STRING` command, like in 'old' BibTeX [16, § 13.2.3]:

```
@STRING{roma = {[Roma] * italian
                [Rome] * english
                [Rom] * german}}
```

the sequence of '`[...] * ...`' notations being a syntactical extension of MlBibTeX for information that *must* be put, possibly in another language if no translation is available [4]. In other words, processing such a sequence never yields an empty string. But this `@STRING` command is expanded *verbatim* when a .bib file is processed—as in 'old' BibTeX—so it must be put in every .bib file where this `roma` symbol is used.

A better solution, from our point of view, is to define this symbol like month names or predefined names of journals within the standard bst styles of BibTeX. If the `roma` symbol is not defined by a `@STRING` command, it is processed as we show in Figure 4 when a .bib file is parsed. It can defined within a bibliography style—that is, within a .nbst file—as follows:

16. TeX provides a powerful framework to format texts nicely, but to be fit for use, the definitions of this framework need to be organised in a *format*. The first formats were *Plain TeX* and LaTeX, another format, more modern, is ConTeXt [1].

```
<nbst:variable name="roma">
  <nonemptyinformation>
    <group language="italian">Roma</group>
    <group language="english">Rome</group>
    <group language="german">Rom</group>
  </nonemptyinformation>
</nbst:variable>
```

These two ways to define a symbol are close together, since a sequence of '`[...] * ...`' yields a `nonemptyinformation` element when .bib files are parsed. However, if an abbreviation has not been defined in 'old' BibTeX, a warning message is issued and the undefined symbol is just replaced by an empty string. As mentioned in § 1.2, the nbst language allows you to check that a symbol has been defined or not.

Acknowledgements

Many thanks to Maurizio W. Himmelmann for his patience when he was waiting for the first version of this article and for its Italian translation of the abstract. I also thank the referee of this article for the improvements suggested to me.

References

[1] Hans Hagen: *ConTeXt, the Manual*. November 2001. `http://www.pragma-ade.com/general/manuals/cont-enp.pdf`.

[2] Jean-Michel Hufflen: "MlBibTeX: a New Implementation of BibTeX". In: Simon Pepping, ed., *EuroTeX 2001*, pp. 74–94. Kerkrade, The Netherlands. September 2001.

[3] Jean-Michel Hufflen: "European Bibliography Styles and MlBibTeX". TUG*boat*, Vol. 24, no. 3, pp. 489–498. EuroTeX 2003, Brest, France. June 2003.

[4] Jean-Michel Hufflen: "MlBibTeX's Version 1.3". TUG*boat*, Vol. 24, no. 2, pp. 249–262. July 2003.

[5] Jean-Michel Hufflen: "MlBibTeX: beyond LaTeX". In: Apostolos Syropoulos, Karl Berry, Yannis Haralambous, Baden Hugues, Steven Peter and John Plaice, eds., *International Conference on TeX, XML, and Digital Typography*, Vol. 3130 of LNCS, pp. 203–215. Springer, Xanthi, Greece. August 2004.

[6] Jean-Michel Hufflen: "Making MlBibTeX Fit for a Particular Language. Example of the Polish Language". *Biuletyn* GUST, Vol. 21, pp. 14–26. 2004.

[7] Jean-Michel Hufflen: *Managing Languages within MlBibTeX*. To appear. June 2005.

[8] Jean-Michel Hufflen: "Implementing a Bibliography Processor in Scheme". In: J. Michael Ashley and Michel Sperber, eds., *Proc. of the 6th Workshop on Scheme and Functional Programming*, Vol. 619 of *Indiana University Computer Science Department*, pp. 77–87. Tallinn. September 2005.

[9] Jean-Michel Hufflen: "MlBibTEX: a Survey". In: *Proc. GUIT Meeting*, pp. 171–179. Pisa,Italy. October 2005.

[10] Jean-Michel Hufflen: "BibTEX, MlBibTEX and Bibliography Styles". *Biuletyn* GUST, Vol. 23, pp. 76–80. In *BachoTEX 2006 conference*. April 2006.

[11] Jean-Michel Hufflen: "MlBibTEX Meets ConTEXt". In: *EuroTEX 2006 conference, preprints*, pp. 71–76. Debrecen, Hungary. July 2006.

[12] Jean-Michel Hufflen: "Implementing a variant of XSLT in Scheme". In: *Proc.* IFL*'06*, pp. 484–491. Budapest, Hungary. September 2006.

[13] Richard Kelsey, William D. Clinger, Jonathan A. Rees, Harold Abelson, Norman I. Adams iv, David H. Bartley, Gary Brooks, R. Kent Dybvig, Daniel P. Friedman, Robert Halstead, Chris Hanson, Christopher T. Haynes, Eugene Edmund Kohlbecker, Jr, Donald Oxley, Kent M. Pitman, Guillermo J. Rozas, Guy Lewis Steele, Jr, Gerald Jay Sussman and Mitchell Wand: "Revised^5 Report on the Algorithmic Language Scheme". HOSC, Vol. 11, no. 1, pp. 7–105. August 1998.

[14] Oleg E. Kiselyov: XML *and Scheme*. September 2005. `http://okmij.org/ftp/Scheme/xml.html`.

[15] Leslie Lamport: *LaTEX: A Document Preparation System. User's Guide and Reference Manual*. Addison-Wesley Publishing Company, Reading, Massachusetts. 1994.

[16] Frank Mittelbach and Michel Goossens, with Johannes Braams, David Carlisle, Chris A. Rowley, Christine Detig and Joachim Schrod: *The LaTEX Companion*. 2nd edition. Addison-Wesley Publishing Company, Reading, Massachusetts. August 2004.

[17] Oren Patashnik: *BibTEXing*. February 1988. Part of the BibTEX distribution.

[18] Erik T. Ray: *Learning* XML. O'Reilly & Associates, Inc. January 2001.

[19] The Unicode Consortium: *The Unicode Standard Version 4.0*. Addison-Wesley. August 2003.

[20] W3C: XML *Path Language (XPath). Version 1.0*. W3C Recommendation. Edited by James Clark and Steve DeRose. November 1999. `http://www.w3.org/TR/1999/REC-xpath-19991116`.

[21] W3C: XSL *Transformations (*XSLT*). Version 1.0*. W3C Recommendation. Edited by James Clark. November 1999. `http://www.w3.org/TR/1999/REC-xslt-19991116`.

▷ Jean-Michel HUFFLEN
LIFC (FRE CNRS 2661)
University of Franche-Comté
16, route de Gray
25030 BESANÇON CEDEX
FRANCE
`hufflen@lifc.univ-fcomte.fr`